A model of guarded recursion with clock synchronisation

Aleš Bizjak [1]

Department of Computer Science, Aarhus University, Denmark

Rasmus Ejlers Møgelberg[2]

IT University of Copenhagen, Denmark

Abstract

Guarded recursion is an approach to solving recursive type equations where the type variable appears guarded by a modality to be thought of as a delay for one time step. Atkey and McBride proposed a calculus in which guarded recursion can be used when programming with coinductive data, allowing productivity to be captured in types. The calculus uses *clocks* representing time streams and *clock quantifiers* which allow limited and controlled elimination of modalities. The calculus has since been extended to dependent types by Møgelberg. Both works give denotational semantics but no rewrite semantics.
In previous versions of this calculus, different clocks represented separate time streams and *clock synchronisation* was prohibited. In this paper we show that allowing clock synchronisation is safe by constructing a new model of guarded recursion and clocks. This result will greatly simplify the type theory by removing freshness restrictions from typing rules, and is a necessary step towards defining rewrite semantics, and ultimately implementing the calculus.

Keywords: Guarded recursion, coinductive types, type theory, categorical semantics.

1 Introduction

Guarded recursion [17] is an approach to solving recursive type equations where the type variable appears *guarded* by a $\blacktriangleright$ (pronounced "later") modal type operator. In particular the type variable could appear positively or negatively or both, e.g. the equation $\sigma = 1 + \blacktriangleright(\sigma \to \sigma)$ has a unique solution [6]. On the term level the *guarded fixed point combinator* $\mathsf{fix}_\tau : (\blacktriangleright\tau \to \tau) \to \tau$ satisfies the equation $f\,(\mathsf{next}\,(\mathsf{fix}_\tau f)) = \mathsf{fix}_\tau f$ for any $f : \blacktriangleright\tau \to \tau$. Here $\mathsf{next} : \tau \to \blacktriangleright\tau$ is an operation that "freezes" an element that we have available now so that it is only available in the next time step.

One situation where guarded recursive types are useful is when faced with an unsolvable type equation. These arise for example when modelling programming

[1] Email: abizjak@cs.au.dk
[2] Email: mogel@itu.dk

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

languages with sophisticated features. In this case a solution to a guarded version of the equation often turns out to suffice, as shown in [6].

But guarded recursive versions of polymorphic type equations are also useful in type theory, even in settings where inductive and coinductive solutions to these equations are assumed to exist. To see this, consider the coinductive type of streams Str, i.e., the final coalgebra for the functor $\mathbb{S}(X) = \mathbb{N} \times X$. Proof assistants like Coq [14] and Agda [18] allow programmers to construct streams using recursive definitions, but to ensure consistency, these must be *productive*, i.e., one must be able to compute the first n elements of a stream in finite time. Coq and Agda inspect recursive definitions for productivity by a *syntactic property* that is often overly conservative and does not interact well with higher-order functions.

Using the type of *guarded streams* Str_g, i.e., the *unique* type satisfying the equation $\mathsf{Str}_\mathsf{g} = \mathbb{N} \times \blacktriangleright \mathsf{Str}_\mathsf{g}$, one can encode productivity in types: a productive recursive stream definition is exactly a term of type $\blacktriangleright \mathsf{Str}_\mathsf{g} \to \mathsf{Str}_\mathsf{g}$. To combine the benefits of coinductive and guarded recursive types, Atkey and McBride [3] suggested a simply typed calculus with clock variables κ representing time streams, each with associated $\blacktriangleright^\kappa$ type constructors, and universal quantification over clocks $\forall\kappa$. If we think of the type τ as being time-indexed along κ, then the type $\forall\kappa.\tau$ contains only elements which are available for all time steps. The relationship between the two notions of streams can then be captured by the encoding of the coinductive stream type as $\mathsf{Str} = \forall\kappa.\, \mathsf{Str}_\mathsf{g}{}^\kappa$. This encoding works for a general class of coinductive types including those given by polynomial functors, and these results were since extended to the dependently typed setting by Møgelberg [16]. In both cases the encodings were proved sound with respect to a denotational model and no rewrite semantics was given. This paper is part of ongoing work to construct just that.

Clock synchronisation

In the calculus for guarded recursion with clocks, typing judgements are given in a context of clocks Δ, which is just a finite set of names for clocks, as well as a context of term variables Γ. Clock variables κ are simply names, there are no constants or operations on them, and there is no type of clocks. The introduction and elimination rules for $\forall\kappa$ as defined by Atkey and McBride [3] are

$$\frac{\Delta, \kappa \mid \Gamma \vdash t : \tau}{\Delta \mid \Gamma \vdash \Lambda\kappa.t : \forall\kappa.\tau} \qquad \frac{\Delta, \kappa' \mid \Gamma \vdash t : \forall\kappa.\tau \qquad \kappa' \notin \forall\kappa.\tau}{\Delta, \kappa' \mid \Gamma \vdash t[\kappa'] : \tau[\kappa'/\kappa]} \tag{1}$$

These rules are very similar to those for polymorphic types in System F [8], except for the freshness side condition on the elimination rule ensuring that the clocks κ and κ' are not synchronised in τ. The side condition makes the rule syntactically not well-behaved. For instance it is not clear that the β-rule for clock application preserves types.

This becomes a more serious problem in dependent type theory. The rule Møgelberg [16] considers for clock instantiation is

$$\frac{\kappa \notin \mathrm{fc}\,(\Gamma) \qquad \Delta, \kappa \mid \Gamma \vdash \tau \qquad \Delta, \kappa' \mid \Gamma, \Gamma' \vdash t : \forall\kappa.\tau}{\Delta, \kappa' \mid \Gamma, \Gamma' \vdash t[\kappa'] : \tau[\kappa'/\kappa]}$$

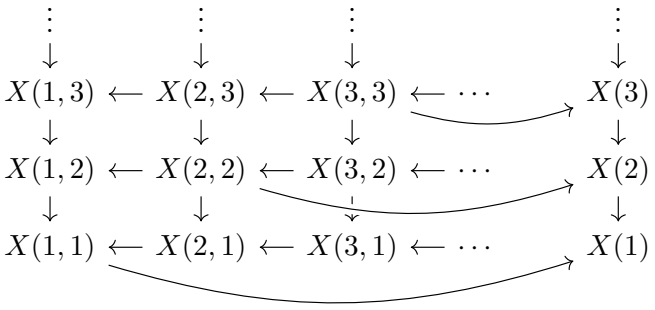


Figure 1. A type with two free clocks in the new model.

where the side condition requires that none of the types τ depends on contain the clock κ. The reason for the additional clock context Γ' is to ensure that the calculus is closed under weakening. However, closure under substitution was overlooked and the rules do not appear to be sufficient to derive the substitution property like

$$\frac{\Delta \mid \Gamma, x : \tau \vdash t : \sigma \qquad \Delta \mid \Gamma \vdash s : \tau}{\Delta \mid \Gamma \vdash t[s/x] : \sigma[s/x]}$$

which is necessary for a well-behaved dependent type theory.

The restriction on clock instantiation comes from the denotational models of guarded recursion. The original work on guarded recursion [5,6] models a type as a presheaf over the ordered natural numbers, i.e., a diagram of the form

$$X(1) \leftarrow X(2) \leftarrow X(3) \leftarrow \cdots$$

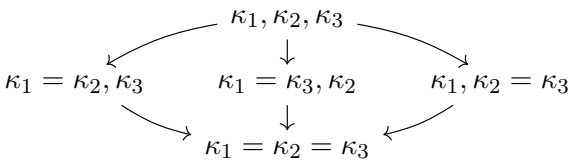
For example, the guarded recursive type of streams satisfying $\mathsf{Str_g} = \mathbb{N} \times \blacktriangleright \mathsf{Str_g}$ is modelled by the presheaf with $X(n) = \mathbb{N}^n$. In this model $\blacktriangleright$ shifts a type one step to the right inserting a singleton set in the end of the sequence.

This model was generalised by Møgelberg [16] (Atkey and McBride [3] use essentially the same idea) to multiple clocks by simply indexing by multiple copies of natural numbers. Thus, conceptually, a type with clocks κ_1 and κ_2 was modelled as a two dimensional diagram of sets (as in the left hand part of Figure 1). In this model there is no semantic correspondent to clock substitution. In particular, if τ is a type with two free clocks κ_1 and κ_2, then the denotation of $\tau[\kappa_1/\kappa_2]$ is a one dimensional diagram, but this is in general not the diagonal of the denotation of τ, as one might expect. Semantically, one reason is that taking the diagonal does not commute with the cartesian closed structure.

We propose a new model which supports clock substitution that preserves all the constructs of type theory in the correct way. The model verifies soundness (up to solving the coherence problem, see Section 4) of the rules (1) as understood in dependent type theory, but without the freshness side condition on the elimination rule. In the new model a type depending on two clocks κ_1 and κ_2 is modelled as a commutative diagrams of the form in Figure 1: the two dimensional grid on the left represents the type X when clocks κ_1 and κ_2 are not identified and the vertical diagram on the right represents the type X when clocks κ_1 and κ_2 become

synchronised. The arrows inside the two and one dimensional diagrams describe the evolution of elements when the clocks decrease and the arrows from the diagonal of the diagram on the left to the diagram on the right describe how the elements change when the clocks are synchronised. This also explains why there are no arrows from the vertical diagram on the right to the diagram on the left. Once the clocks are identified there is no way to disentangle them. To model the substitution κ_1/κ_2 we simply take the right vertical part of the diagram.

With more clocks the denotation of a type becomes more complex. For instance when we have three clocks the denotation will have a three dimensional diagram (representing the state when none of the clocks are identified), three two dimensional diagrams (representing the state when two of the clocks are identified) and a one dimensional diagram, representing the state when all of the clocks are identified. Arrows between the different diagrams are given according to the following schema



where, for example, $\kappa_1 = \kappa_2, \kappa_3$ represents the diagram where clocks κ_1 and κ_2 are identified, and κ_3 is independent of the two.

Related work

The calculus considered in this paper can be understood as a modal variant of sized types [1,2]. The modal aspect of $\forall\kappa$ is investigated by Clouston *et. al.* [7] which replaces clocks and quantification $\forall\kappa$ by a single comonadic modality $\blacksquare$. This corresponds to having exactly one clock always available. And indeed the calculus is modelled in the topos of trees. The paper provides operational semantics for the calculus and a logic, which is essentially the internal language of the topos of trees with some additional constructs and rules, for reasoning about equality of programs.

2 Rules of the type theory

Due to space restrictions we only give a brief overview of some of the type and term constructs which are not part of basic dependent type theory. For details on how to use the terms we refer to Møgelberg [16] and Atkey and McBride [3].

The new types in addition to standard constructs of dependent type theory are

$$\frac{\Delta \mid \Gamma \vdash \tau}{\Delta \mid \Gamma \vdash \blacktriangleright^\kappa \tau}\ \kappa \in \Delta \qquad \frac{\Delta \vdash \Gamma \qquad \Delta, \kappa \mid \Gamma \vdash \tau}{\Delta \mid \Gamma \vdash \forall\kappa.\tau}\ \kappa \notin \Delta \qquad \frac{\Delta' \subseteq \Delta \qquad \Delta \vdash \Gamma}{\Delta \mid \Gamma \vdash \mathcal{U}_{\Delta'}}$$

The first two rules introduce $\blacktriangleright^\kappa$ and $\forall\kappa.$ type formers. The third rule gives universes. The reason we need universes $\mathcal{U}_{\Delta'}$ for each $\Delta' \subseteq \Delta$ is to ensure that they are preserved by clock substitution, in particular by weakening. Clock substitution from clock context Δ_1 to clock context Δ_2 is given by a function $f : \Delta_1 \to \Delta_2$, e.g. a substitution κ_1/κ_2 from clock context κ_1, κ_2 to clock context κ_1 is given by the

unique function. We point out in particular how clock substitution on universes is defined, on other constructs it is standard. If $f : \Delta_1 \to \Delta_2$ is a clock substitution and $\Delta' \subseteq \Delta_1$ then we define $f(\mathcal{U}_{\Delta'}) = \mathcal{U}_{f[\Delta']}$, where $f[\Delta']$ denotes the image of the set Δ' by f. For example $(\mathcal{U}_{\kappa_1,\kappa_2})\,[\kappa_1/\kappa_2] = \mathcal{U}_{\kappa_1}$. Note that this would not make sense if we only had one universe $\mathcal{U}_\Delta$ in each clock context Δ, since f might not be surjective. See Section 3.5 and also Møgelberg [16] for semantic reasons why these additional universes are needed.

The main terms introducing and eliminating the new constructs are

$$\frac{\Delta \mid \Gamma \vdash t : \tau}{\Delta \mid \Gamma \vdash \mathsf{next}^\kappa\, t : \blacktriangleright^\kappa \tau}\ \kappa \in \Delta \qquad \frac{\Delta \mid \Gamma \vdash t : \blacktriangleright^\kappa(\tau \to \sigma) \qquad \Delta \mid \Gamma \vdash s : \blacktriangleright^\kappa \tau}{\Delta \mid \Gamma \vdash t \circledast^\kappa s : \blacktriangleright^\kappa \sigma}$$

$$\frac{\Delta \vdash \Gamma \qquad \Delta, \kappa \mid \Gamma \vdash t : \tau}{\Delta \mid \Gamma \vdash \Lambda\kappa.t : \forall\kappa.\tau}\ \kappa \notin \Delta \qquad \frac{\Delta \mid \Gamma \vdash t : \forall\kappa.\tau}{\Delta \mid \Gamma \vdash t[\kappa'] : \tau\,[\kappa'/\kappa]}\ \kappa' \in \Delta$$

$$\frac{\Delta \mid \Gamma, x : \blacktriangleright^\kappa \tau \vdash t : \tau}{\Delta \mid \Gamma \vdash \mathsf{fix}^\kappa\, x.t : \tau}\ \kappa \in \Delta$$

The constructs next^κ and $\circledast^\kappa$ are part of the applicative functor [15] structure of $\blacktriangleright^\kappa$. The second line contains introduction and elimination forms for the $\forall\kappa$ type. The term $\mathsf{fix}^\kappa\, x.t$ is *the unique* fixed point of t.

In addition to standard rules these constructs satisfy type isomorphisms

$$\tau \cong \forall\kappa.\tau \quad \text{if } \kappa \notin \tau \qquad \forall\kappa.\tau + \forall\kappa.\sigma \cong \forall\kappa.(\tau + \sigma)$$

$$\sum_{x:\tau} \forall\kappa.\sigma \cong \forall\kappa. \sum_{x:\tau} \sigma \quad \text{if } \kappa \notin \tau \qquad \blacktriangleright^{\kappa'} \forall\kappa.\tau \cong \forall\kappa.\, \blacktriangleright^{\kappa'} \tau \quad \text{for } \kappa \neq \kappa'$$

$$\forall\kappa.\tau \cong \forall\kappa.\, \blacktriangleright^\kappa \tau$$

which are needed for encoding coinductive types using guarded recursive types. The directions from left to right are definable in the calculus with only the standard introduction and elimination forms, but the inverses need to be added as additional terms, together with definitional equalities stating that they are inverses. Møgelberg [16] explains in detail how this is done.

3 The new model

We fix a countable set of clocks $\mathrm{CV} = \{\kappa_1, \kappa_2, \ldots\}$. The model we construct can be briefly described as follows. We build an indexed category $\mathfrak{GR}$, indexed by the opposite of the full subcategory of Set on *finite subsets* of CV. For each finite set of clocks Δ, the category $\mathfrak{GR}(\Delta)$ is a model of extensional dependent type theory: term variable contexts $\Delta \vdash \Gamma$, types $\Delta \mid \Gamma \vdash A$ and terms $\Delta \mid \Gamma \vdash t : A$ are interpreted in $\mathfrak{GR}(\Delta)$. For any $f : \Delta_1 \to \Delta_2$ the reindexing functor $\mathfrak{GR}(f) : \mathfrak{GR}(\Delta_1) \to \mathfrak{GR}(\Delta_2)$, which is used to model clock substitution, preserves all the structure required for modeling dependent type theory. Finally, for any inclusion $\iota : \Delta \to \Delta, \kappa$ the reindexing functor $\mathfrak{GR}(\iota) : \mathfrak{GR}(\Delta) \to \mathfrak{GR}(\Delta, \kappa)$ has a right adjoint $\forall\kappa$ which is used to interpret quantification over clocks. Due to space restrictions we cannot describe the model in whole, but we only provide definitions

of constructs used to interpret $\blacktriangleright^\kappa$, $\forall\kappa$ and the universes and proof sketches of important points.

3.1 The indexed category $\mathfrak{GR}$

The category $\mathfrak{GR}(\Delta)$ is the category of presheaves over the poset $\mathfrak{I}(\Delta)$ which we describe first. To understand the definition of the poset $\mathfrak{I}(\Delta)$ it is useful to keep in mind the example in Figure 1. Let Δ be a finite set of clocks. An element $\mathfrak{I}(\Delta)$ should indicate what is the state of clocks, i.e. which clock are identified, and it should indicate how much time is left on each clock. Hence elements of $\mathfrak{I}(\Delta)$ should be pairs (E, δ) of an equivalence relation E on Δ and a function $\delta : \Delta \to \mathbb{N}$. Since identified clocks should have the same amount of time remaining, the function δ should preserve E. The order on $\mathfrak{I}(\Delta)$ should allow us to get from state represented by (E, δ) to (E', δ') whenever E' identifies more clocks than E and there is no more time left on δ' than on δ. This makes sense because we want to be able to substitute clocks, and substitution, in general, identifies clocks. On the other hand once the clocks are identified we can no longer separate them, hence we should not be able to get from a state where more clocks are identified to a state where fewer of them are. With this in mind, here are the precise definitions.

Definition 3.1 For $\Delta \subseteq^{\text{fin}} \mathrm{CV}$ let $\mathfrak{E}(\Delta)$ be the set of equivalence relations on Δ (considered as subsets of $\Delta \times \Delta$).

The order relation on $\mathfrak{E}(\Delta)$ is the *opposite of the refinement order*, concretely $E \geq E' \iff E \subseteq E'$ (note the reverse inclusion). Or in other words, $E' \leq E$ if whenever two elements are related by E, they are also related by E'.

The top element for this ordering is the diagonal relation $\mathfrak{d}_\Delta$. The bottom element is the relation that equates everything.

For a function $f : \Delta_1 \to \Delta_2$ let $\mathfrak{E}(f) : \mathfrak{E}(\Delta_2) \to \mathfrak{E}(\Delta_1)$ be the function defined by pullback as $\mathfrak{E}(f)(E) = \{(\kappa_1, \kappa_2) \mid (f(\kappa_1), f(\kappa_2)) \in E\}$, i.e. clocks κ_1 and κ_2 are related by $\mathfrak{E}(f)(E)$ if they become equated in E after substitution with f.

Definition 3.2 Let Δ be a finite set of clocks. The poset $\mathfrak{I}(\Delta)$ has elements pairs (E, δ) where $E \in \mathfrak{E}(\Delta)$ is an equivalence relation and $\delta : \Delta \to \mathbb{N}$ is a function that respects E. This means that if $(\kappa_1, \kappa_2) \in E$ then $\delta(\kappa_1) = \delta(\kappa_2)$.

The order on $\mathfrak{I}(\Delta)$ is component-wise: $(E, \delta) \geq (E', \delta') \iff E \geq E' \wedge \delta \geq \delta'$. where the ordering on functions is pointwise.

For a function $f : \Delta_1 \to \Delta_2$ the function $\mathfrak{I}(f) : \mathfrak{I}(\Delta_2) \to \mathfrak{I}(\Delta_1)$ is defined as $\mathfrak{I}(f)(E, \delta) = (\mathfrak{E}(f)(E), \delta \circ f)$.

Definition 3.3 Let Δ be a finite set of clocks. The category $\mathfrak{GR}(\Delta)$ is the category $\mathrm{Set}^{(\mathfrak{I}(\Delta))^{\mathrm{op}}}$ of (contravariant) $\mathfrak{I}(\Delta)$-indexed set valued presheaves.

For a function $f : \Delta_1 \to \Delta_2$ let $\mathfrak{GR}(f) : \mathfrak{GR}(\Delta_1) \to \mathfrak{GR}(\Delta_2)$ be the functor defined by precomposition with $\mathfrak{I}(f)$. Concretely

$$\mathfrak{GR}(f)(X) = X \circ \mathfrak{I}(f) \qquad \text{and} \qquad \mathfrak{GR}(f)(\alpha)_{(E,\delta)} = \alpha_{\mathfrak{I}(f)(E,\delta)}$$

where X is an object of $\mathfrak{GR}(\Delta_1)$, α is a natural transformation in $\mathfrak{GR}(\Delta_1)$ and $(E, \delta) \in \mathfrak{I}(\Delta_2)$. We will also use f^* for the functor $\mathfrak{GR}(f)$.

For use in Section 3.5 below we record a property of surjective substitutions.

Lemma 3.4 *Let $f : \Delta_1 \to \Delta_2$ be a function between clock contexts. If f is* surjective *then $\mathfrak{E}(f)$ and $\mathfrak{I}(f)$ are* injective.

3.2 Basic properties of $\mathfrak{GR}$

For each finite set of clocks the category $\mathfrak{GR}(\Delta)$ is a presheaf topos, hence it is a model of extensional dependent type theory. As mentioned above we aim to use the functors $\mathfrak{GR}(f)$ to interpret clock substitution and this means that these functors must preserve constructs used to interpret dependent type theory.

The first property we show is that all the functors $\mathfrak{GR}(f)$ are locally cartesian closed functors. This property is not so straightforward to show and requires some preparations. First, because the functors $\mathfrak{GR}(f)$ are given by precomposition, they have left and right adjoints [13, Theorem VII.2.2]. Hence they preserve all limits and colimits and in fact they preserve the natural choice of these on the nose, a property that simplifies some proofs. To show that they also preserve exponentials and local exponentials we require some preparation.

Definition 3.5 Let P and Q be two posets. An order-preserving function $\phi : P \to Q$ is a *fibration* if for every $p \in P$ and $q \in Q$ such that $q \leq \phi(p)$ the set

$$B_{p,q} = \{p' \leq p \mid \phi(p') = q\}$$

has a *top* element $u(p,q)$ and moreover whenever $q_1 \leq q_2$, also $u(p,q_1) \leq u(p,q_2)$.

This definition is equivalent to a standard definition of a fibration [10], but we found it useful to have names for the top element $u(p,q)$.

One of the reasons *fibrations* are useful is the following property.

Proposition 3.6 *Let P and Q be two posets and $\phi : P \to Q$* a fibration. *The functor $\phi^* : \mathrm{Set}^{Q^{op}} \to \mathrm{Set}^{P^{op}}$ given by precomposition with ϕ, i.e. $\phi^*(X) = X \circ \phi$, is a* locally cartesian closed functor.

Proof sketch It is possible to show this directly, but ϕ being a fibration implies the assumption of Lemma C.3.3.8.(*ii*) of Johnstone [11] which shows in particular that the functor ϕ^* is locally cartesian closed by Proposition C.3.3.1 of *loc. cit.* □

Next, we show the crucial property in detail.

Lemma 3.7 *Let Δ_1, Δ_2 be two finite sets of clocks and $f : \Delta_1 \to \Delta_2$ a function. Then $\mathfrak{E}(f) : \mathfrak{E}(\Delta_2) \to \mathfrak{E}(\Delta_1)$ and $\mathfrak{I}(f) : \mathfrak{I}(\Delta_2) \to \mathfrak{I}(\Delta_1)$ are both* fibrations.

Proof

$\mathfrak{E}(f)$ is a fibration Let $E \in \mathfrak{E}(\Delta_2)$ and $\mathfrak{E}(\Delta_1) \ni F \leq \mathfrak{E}(f)(E)$. Define $u(E,F) \in \mathfrak{E}(\Delta_2)$ as the *transitive closure* of the relation

$$E_b = \{(\kappa,\kappa') \mid (\kappa,\kappa') \in E \vee (\exists(\kappa_1,\kappa_2) \in F, f(\kappa_1) = \kappa \wedge f(\kappa_2) = \kappa')\}$$

The relation E_b is reflexive because E is and it is symmetric because E and F are symmetric. The transitive closure of a reflexive and symmetric relation is

again reflexive and symmetric and by definition also transitive. Hence $u(E,F)$ is an equivalence relation. The first part of the disjunction in the definition of E_b ensures $u(E,F) \leq E$.

Next we check that $\mathfrak{E}(f)(u(E,F)) = F$ by showing two inclusions.

- First the easy direction. Take $(\kappa_1,\kappa_2) \in F$. We need to show that $(f(\kappa_1),f(\kappa_2)) \in u(E,F)$. This is simple because $(f(\kappa_1),f(\kappa_2)) \in E_b$ since it satisfies the second part of the defining condition by choosing witnesses κ_1 and κ_2.
- The converse inclusion is more involved. First we show that if $(\kappa_1,\kappa_2) \in \mathfrak{E}(f)(E_b)$, meaning $(f(\kappa_1),f(\kappa_2)) \in E_b$, then $(\kappa_1,\kappa_2) \in F$. So let $(f(\kappa_1),f(\kappa_2)) \in E_b$. Then
  - either $(f(\kappa_1),f(\kappa_2)) \in E$ in which case $(\kappa_1,\kappa_2) \in \mathfrak{E}(f)(E)$ and so $(\kappa_1,\kappa_2) \in F$ (because $F \leq \mathfrak{E}(f)(E)$)
  - or there are $(\kappa_1',\kappa_2') \in F$ such that $f(\kappa_1') = f(\kappa_1)$ and $f(\kappa_2') = f(\kappa_2)$. Because E is reflexive and $F \leq \mathfrak{E}(f)(E)$ we have $(\kappa_2',\kappa_2) \in F$ and $(\kappa_1,\kappa_1') \in F$. Using transitivity of F we get $(\kappa_1,\kappa_2) \in F$.

  To conclude we show that $\mathfrak{E}(f)(E_b) \supseteq \mathfrak{E}(f)(E_b \circ E_b)$ where $E_b \circ E_b$ is composition of relations. Because E_b is reflexive and $\mathfrak{E}(f)$ monotone this implies $\mathfrak{E}(f)(E_b) = \mathfrak{E}(f)(E_b \circ E_b)$. Finally because $\mathfrak{E}(f)$ commutes with unions, which is easy to check directly from the definition of $\mathfrak{E}(f)$, this result implies $\mathfrak{E}(f)(u(E,F)) = \mathfrak{E}(f)(E_b) \subseteq F$. The last inclusion is what we have shown above.

  So take $(\kappa_1,\kappa_2) \in \mathfrak{E}(f)(E_b \circ E_b)$. By definition $(f(\kappa_1),f(\kappa_2)) \in E_b \circ E_b$ so there is a z, such that $(f(\kappa_1),z) \in E_b$ and $(z,f(\kappa_2)) \in E_b$.
  - If $(f(\kappa_1),z) \in E$ and $(z,f(\kappa_2)) \in E$ then by transitivity of E also $(f(\kappa_1),f(\kappa_2)) \in E$ and so $(\kappa_1,\kappa_2) \in \mathfrak{E}(f)(E_b)$.
  - Otherwise $z = f(\kappa)$ for some κ such that $(\kappa_1,\kappa) \in F$ or $(\kappa,\kappa_2) \in F$. The cases are symmetric because E_b is so we only consider the case when $(\kappa_1,\kappa) \in F$. Observe that in such a case we also have $(\kappa,\kappa_2) \in F$. Indeed, if $(z,f(\kappa_2)) \in E_b$ then either $(z,f(\kappa_2)) \in E$ in which case we have $(\kappa,\kappa_2) \in F$ from the assumption $F \leq \mathfrak{E}(f)(E)$, or there is a pair $(\kappa',\kappa_2') \in F$ such that $f(\kappa') = z = f(\kappa)$ and $f(\kappa_2') = f(\kappa_2)$. Because E is reflexive and $F \leq \mathfrak{E}(f)(E)$ we have $(\kappa',\kappa) \in F$ and $(\kappa_2',\kappa_2) \in F$. Thus by transitivity and symmetry of F we have $(\kappa,\kappa_2) \in F$. This further gives $(\kappa_1,\kappa_2) \in F$ which shows $(f(\kappa_1),f(\kappa_2)) \in E_b$, concluding the proof.

  To see that $u(E,F)$ is the largest $E' \leq E$ such that $\mathfrak{E}(f)(E') = F$ take some E' satisfying this condition and observe that it suffices to show $E_b \subseteq E'$ because E' is transitive. So take $(\kappa,\kappa') \in E_b$. If $(\kappa,\kappa') \in E$ then $(\kappa,\kappa') \in E'$ by using $E' \leq E$. On the other hand if there are $(\kappa_1,\kappa_2) \in F$ such that $f(\kappa_1) = \kappa$ and $f(\kappa_2) = \kappa'$ then by definition we have $(\kappa_1,\kappa_2) \in F = \mathfrak{E}(f)(E')$. Hence $(f(\kappa_1),f(\kappa_2)) \in E'$ and thus $(\kappa,\kappa') \in E'$.

  The last property to check is that if $F_1 \leq F_2 \leq \mathfrak{E}(f)(E)$ then $u(E,F_1) \leq u(E,F_2)$. This is immediate from the explicit definition of the relations E_b.

$\mathfrak{I}(f)$ is a fibration To see that $\mathfrak{I}(f)$ is a fibration let $(E,\delta) \in \mathfrak{I}(\Delta_2)$ and $\mathfrak{I}(\Delta_1) \ni$

$(F,\gamma) \leq \mathfrak{I}(f)(E,\delta)$. Define $\delta' : \Delta_2 \to \mathbb{N}$ as

$$\delta'(\kappa) = \begin{cases} \gamma(\kappa_1) & \text{if } \exists \kappa_1 \in \Delta_1, (\kappa, f(\kappa_1)) \in E \\ \delta(\kappa) & \text{otherwise} \end{cases}$$

Then define $u((E,\delta),(F,\gamma)) = (u(E,F),\delta')$ where $u(E,F)$ is the element given by the first part of the proof.

First we check that in the first case it does not matter which κ_1 we choose, i.e., that δ' is well-defined. Suppose $(\kappa, f(\kappa_1)) \in E$ and $(\kappa, f(\kappa_2)) \in E$. Then $(f(\kappa_1), f(\kappa_2)) \in E$ and so $(\kappa_1,\kappa_2) \in \mathfrak{E}(f)(E) \subseteq F$. Hence $\gamma(\kappa_1) = \gamma(\kappa_2)$ because γ respects F.

Now to show that $(u(E,F),\delta')$ is an element of $\mathfrak{I}(\Delta_2)$ we need to show that δ' respects $u(E,F)$. Because equality is transitive it suffices to check that if $(\kappa,\kappa') \in E_b$ then $\delta'(\kappa) = \delta'(\kappa')$. So take such κ,κ'. We consider two cases:

- there exist $(\kappa_1,\kappa_2) \in F$ such that $f(\kappa_1) = \kappa$ and $f(\kappa_2) = \kappa'$. Then because E is reflexive the first case of the definition of δ' applies and since $\gamma(\kappa_1) = \gamma(\kappa_2)$, because γ respects F, we also have $\delta'(\kappa) = \delta'(\kappa')$.
- the second case is when $(\kappa,\kappa') \in E$. We split into two further cases.
  - If there is a $\kappa_1 \in \Delta_1$ such that $(\kappa, f(\kappa_1)) \in E$ then we also have $(\kappa', f(\kappa_1)) \in E$ because of symmetry and transitivity of E and so $\delta'(\kappa) = \gamma(\kappa_1) = \delta'(\kappa)$.
  - Otherwise the other case applies and we use the fact that δ preserves E.

Thus we have shown that δ' well-defined. Now observe that because E is reflexive we have $\delta'(f(\kappa)) = \gamma(\kappa)$ hence we have $\mathfrak{I}(f)(u(E,F),\delta') = (F,\gamma)$.

To see that $\delta' \leq \delta$ let $\kappa \in \Delta_2$ and we consider two cases:

- if $(\kappa, f(\kappa_1)) \in E$ for some $\kappa_1 \in \Delta_1$. Then because $\gamma \leq \delta \circ f$ and δ preserves E we have $\delta(\kappa) = \delta(f(\kappa_1)) \geq \gamma(\kappa_1) = \delta'(\kappa)$.
- otherwise $\delta(\kappa) = \delta'(\kappa)$.

In both cases we have $\delta(\kappa) \geq \delta'(\kappa)$ so we conclude $\delta \geq \delta'$.

Suppose now that (E'',δ'') is such that $\mathfrak{I}(f)(E'',\delta'') = (F,\gamma)$ and $(E'',\delta'') \leq (E,\delta)$. Then we know from the first part of the this proof that $E'' \leq u(E,F)$. To see $\delta' \geq \delta''$ take $\kappa \in \Delta_2$ and we consider two cases.

- If $(\kappa, f(\kappa_1)) \in E$ for some $\kappa_1 \in \Delta_1$. Then $\delta'(\kappa) = \gamma(\kappa_1)$. On the other hand $\delta''(\kappa) = \delta''(f(\kappa_1)) = \gamma(\kappa_1)$ which follows from the fact that $\delta'' \circ f = \gamma$.
- Otherwise $\delta'(\kappa) = \delta(\kappa)$ and since $\delta'' \leq \delta$ we have $\delta''(\kappa) \leq \delta(\kappa)$.

In both cases we have $\delta''(\kappa) \leq \delta'(\kappa)$ so we conclude $\delta'' \leq \delta'$ which is what we need.

The fact that the assignment $u((E,\delta),(F,\gamma))$ is order preserving in the second argument follows directly from the definition of δ'.

□

The last two results combined prove the following.

Theorem 3.8 *Let $f : \Delta_1 \to \Delta_2$ be a function between clock contexts. The functor $\mathfrak{GR}(f)$ is a locally cartesian closed functor.*

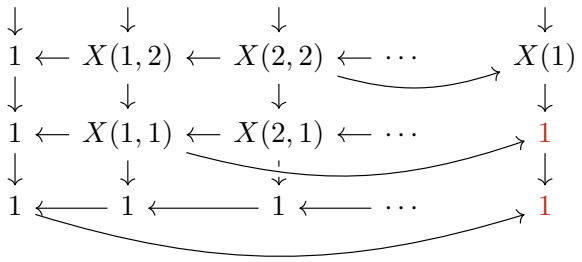
Remark 3.9 As we mentioned already the functors $\mathfrak{GR}(f)$ do preserve the natural choice of limits and colimits on the nose. However there does not appear to be a natural choice of exponentials or dependent products such that $\mathfrak{GR}(f)$ would

preserve them on the nose. As a consequence we have some technical problems with coherence, which we comment on in Section 4 below.

3.3 The $\blacktriangleright^\kappa$ functors

Let Δ be a clock context and $\kappa \in \Delta$. We now define the functor $\blacktriangleright^\kappa$ on $\mathfrak{GR}(\Delta)$ and the natural transformation $\mathsf{next}^\kappa : \mathrm{id}_{\mathfrak{GR}(\Delta)} \to \blacktriangleright^\kappa$ such that the triple $(\mathfrak{GR}(\Delta), \blacktriangleright^\kappa, \mathsf{next}^\kappa)$ is a model of guarded recursive terms [6, Definition 6.1].

Example 3.10 To understand the definition recall the diagram X with two clocks in Figure 1. We wish clock substitution to preserve $\blacktriangleright$ in the sense that $(\blacktriangleright^{\kappa_1} \blacktriangleright^{\kappa_2} X)[\kappa_1/\kappa_2]$ is the same as $\blacktriangleright^{\kappa_1} \blacktriangleright^{\kappa_1} (X[\kappa_1/\kappa_2])$ and so the diagram $\blacktriangleright^{\kappa_1} \blacktriangleright^{\kappa_2} X$ should be



In particular notice that the one dimensional diagram on the left is delayed twice, because it represents the state when κ_1 and κ_2 are identified.

To define $\blacktriangleright^\kappa$ in general we start with an auxiliary definition.

Definition 3.11 Let $\kappa \in \Delta \subseteq^{\mathrm{fin}} \mathrm{CV}$, $E \in \mathfrak{E}(\Delta)$ and $\delta : \Delta \to \mathbb{N}$. The function $\delta^{-\kappa} : \Delta \to \mathbb{N}$ is defined as

$$\delta^{-\kappa}(\kappa') = \begin{cases} \mathsf{max}\{1, \delta(\kappa) - 1\} & \text{if } (\kappa, \kappa') \in E \\ \delta(\kappa') & \text{otherwise} \end{cases}$$

The thing to notice in this definition is that all the clocks equivalent to κ have their remaining time decreased by 1. This is crucial for clock substitution to commute with $\blacktriangleright^\kappa$ in the appropriate way, as illustrated in Example 3.10 above. Decreasing the value of all the clocks related to κ also ensures that if δ preserves E then so does $\delta^{-\kappa}$. This implies $(E, \delta^{-\kappa}) \in \mathfrak{I}(\Delta)$. Observe that $(E, \delta^{-\kappa}) \leq (E, \delta)$ and this assignment is also order preserving. Moreover, this assignment commutes with reindexing $\mathfrak{I}$ as stated in the following lemma.

Lemma 3.12 *Let $f : \Delta_1 \to \Delta_2$ be a function and $(E, \delta) \in \mathfrak{I}(\Delta_2)$. For any $\kappa \in \Delta_1$ the pairs $(\mathfrak{E}(f)(E), \delta^{-f(\kappa)} \circ f)$ and $(\mathfrak{E}(f)(E), (\delta \circ f)^{-\kappa})$ are in $\mathfrak{I}(\Delta_1)$ and moreover they are equal.*

The definition of $\blacktriangleright^\kappa : \mathfrak{GR}(\Delta) \to \mathfrak{GR}(\Delta)$ is now simple.

Definition 3.13 Let $\kappa \in \Delta \subseteq^{\text{fin}} \mathrm{CV}$ and X an object of $\mathfrak{GR}(\Delta)$. The action of the functor $\blacktriangleright^\kappa$ on objects is

$$\blacktriangleright^\kappa(X)(E,\delta) = \begin{cases} 1 & \text{if } \delta(\kappa) = 1 \\ X\left(E, \delta^{-\kappa}\right) & \text{otherwise} \end{cases}$$

$$\blacktriangleright^\kappa(X)\left((E_1,\delta_1) \leq (E_2,\delta_2)\right) = \begin{cases} ! & \text{if } \delta_1(\kappa) = 1 \\ X\left((E_1,\delta_1^{-\kappa}) \leq \left(E_2,\delta_2^{-\kappa}\right)\right) & \text{otherwise} \end{cases}$$

where 1 is the singleton set $\{*\}$ and $!$ is the unique arrow to 1. On morphisms

$$\blacktriangleright^\kappa(\alpha)_{E,\delta} = \begin{cases} \mathrm{id}_1 & \text{if } \delta(\kappa) = 1 \\ \alpha_{E,\delta^{-\kappa}} & \text{otherwise} \end{cases}$$

There is an associated natural transformation $\mathsf{next}^\kappa : \mathrm{id}_{\mathfrak{GR}(\Delta)} \to \blacktriangleright^\kappa$

$$\mathsf{next}^\kappa_{X(E,\delta)}(x) = \begin{cases} * & \text{if } \delta(\kappa) = 1 \\ X\left((E,\delta^{-\kappa}) \leq (E,\delta)\right)(x) & \text{otherwise} \end{cases}$$

It is easy to see that $\blacktriangleright^\kappa$ preserves all limits, since these are given pointwise and any limit of any diagram of terminal objects is a terminal object. It does not preserve colimits, however. For example it does not preserve the initial object.

Proposition 3.14 (Properties of $\blacktriangleright$) *Let Δ_1 and Δ_2 be two clock contexts and $f : \Delta_1 \to \Delta_2$ a function between them. Let $\kappa \in \Delta_1$ be a clock. The following properties hold.*

(i) *Let X, Y be two objects in $\mathfrak{GR}(\Delta_1)$ and $\alpha : Y \times \blacktriangleright^\kappa(X) \to X$ a natural transformation. There exists* a unique *$\beta : Y \to X$ such that $\alpha \circ \langle id_Y, \mathsf{next}^\kappa \circ \beta\rangle = \beta$. We write $\mathsf{fix}^\kappa(\alpha)$ for this unique fixed point. Moreover, for any $\gamma : Z \to Y$ $\mathsf{fix}^\kappa(\alpha) \circ \gamma = \mathsf{fix}^\kappa(\alpha \circ \gamma \times id_Y)$ which expresses naturality of fixed points.*

(ii) *Clock substitution preserves $\blacktriangleright$, i.e. $f^* \circ \blacktriangleright^\kappa = \blacktriangleright^{f(\kappa)} \circ f^*$, and for every $X \in \mathfrak{GR}(\Delta)$, $f^*(\mathsf{next}^\kappa_X) = \mathsf{next}^{f(\kappa)}_{f^*(X)}$.*

(iii) *Let $\alpha : Y \times \blacktriangleright^\kappa X \to X$ be a morphism in $\mathfrak{GR}(\Delta_1)$. From the fact that f^* preserves products on the nose and the previous item the morphism $f^*(\alpha)$ has type $f^*(Y) \times \blacktriangleright^{f(\kappa)} f^*(X) \to f^*(X)$ and moreover $f^*(\mathsf{fix}^\kappa(\alpha)) = \mathsf{fix}^{f(\kappa)}(f^*(\alpha))$.*

Proof sketch The fixed point β at (E,δ) is defined by induction on $\delta(\kappa)$ as

$$\beta_{E,\delta}(y) = \begin{cases} \alpha_{E,\delta}(y,*) & \text{if } \delta(\kappa) = 1 \\ \alpha_{E,\delta}\left(y, \beta_{E,\delta^{-\kappa}}\left(Y\left((E,\delta^{-\kappa}) \leq (E,\delta)\right)(y)\right)\right) & \text{otherwise} \end{cases}$$

Item (ii) is shown by simple unfolding of definitions. Item (iii) is shown by establishing that the term on the left is a fixed point of $f^*(\alpha)$ and then using uniqueness of fixed points. □

The facts above show that for each clock context Δ and $\kappa \in \Delta$, the triple $(\mathfrak{GR}(\Delta), \blacktriangleright^\kappa, \mathsf{next}^\kappa)$ is a model of guarded recursive terms [6, Definition 6.1]. Hence

for each object $X \in \mathfrak{GR}(\Delta)$ the slice category $\mathfrak{GR}(\Delta)/X$ also admits a $\blacktriangleright_\kappa^X$ functor defined by pullback [6, Theorem 6.3]

$$\begin{array}{ccc} \blacktriangleright_\kappa^X Y & \longrightarrow & \blacktriangleright^\kappa Y \\ {\scriptstyle \blacktriangleright_\kappa^X \alpha}\downarrow & & \downarrow{\scriptstyle \blacktriangleright^\kappa \alpha} \\ X & \xrightarrow{\mathsf{next}^\kappa} & \blacktriangleright^\kappa X \end{array}$$

This comes with the associated morphism $\mathsf{next}^{\kappa,X}$ in $\mathfrak{GR}(\Delta)/X$. Moreover, for $f : \Delta \to \Delta'$ we easily conclude from Proposition 3.14 and the fact that the functor f^* preserves all limits on the nose that

$$f^*\left(\blacktriangleright_\kappa^X Y\right) = \blacktriangleright_{f(\kappa)}^{f^*(X)} f^*(Y)$$

and similarly for $f^*\left(\mathsf{next}^{\kappa,X}\right)$ so clock substitution behaves well also with respect to $\blacktriangleright^\kappa$ and next^κ in slices.

3.4 Clock quantification

For any clock context Δ and clock $\kappa \notin \Delta$ the inclusion function $\iota : \Delta \to \Delta, \kappa$ gives rise to the *weakening* functor $\iota^* : \mathfrak{GR}(\Delta) \to \mathfrak{GR}(\Delta, \kappa)$. Because ι^* is defined by precomposition with $\mathfrak{I}(\iota)$ it has a right (as well as left) adjoint [13, Theorem VII.2.2]. We shall call this right adjoint $\forall\kappa$ and in this section we provide a more explicit description of it, which will provide some more intuition behind it and its relation to coinductive types.

To understand the definition it is again useful to consider the case with two clocks from Figure 1. The object $\forall\kappa_2.X$ is a one dimensional diagram and at stage n it is the limit (in Set) of the diagram

$$X(n,1) \leftarrow X(n,2) \leftarrow X(n,3) \leftarrow X(n,4) \leftarrow \cdots$$

The idea is that the type $(\forall\kappa_2.X)(n)$ contains information about $X(n,k)$ for all times k. Note that in particular the one dimensional diagram which represents the state of X when the clocks κ_1 and κ_2 are identified is ignored. This is because the clock κ_2 is no longer free and no substitution will be able to equate it to some other clock, i.e. substitution is capture avoiding.

To define the right adjoint of the inclusion in general we need some auxiliaries.

Lemma 3.15 *Let Δ be a clock context and $\iota : \Delta \to \Delta, \kappa$ the inclusion. Then $\mathfrak{E}(\iota) : \mathfrak{E}(\Delta, \kappa) \to \mathfrak{E}(\Delta)$ has a* right adjoint *$\iota^!$ defined explicitly as*

$$\iota^!(E) = E \cup \{(\kappa, \kappa)\}.$$

In contrast the function $\mathfrak{I}(\iota)$ does not have a right adjoint, the reason being that $\mathbb{N}$ does not have a top element. However for each $n \in \mathbb{N}$ we can define a function $\iota_n^!$

$$\begin{array}{r} \iota_n^! : \mathfrak{I}(\Delta, \kappa) \to \mathfrak{I}(\Delta) \\ \iota_n^!(E, \delta) = (\iota^!(E), \delta_n^!) \end{array} \qquad \text{where} \qquad \delta_n^!(\kappa') = \begin{cases} \delta(\kappa') & \text{if } \kappa' \in \Delta \\ n & \text{if } \kappa' = \kappa \end{cases}$$

Using the explicit description of $\iota^!$ in Lemma 3.15 it is easy to see that $\delta_n^!$ preserves $\iota^!(E)$. We record some useful properties for use below.

Lemma 3.16 *Let Δ be a clock context, $\kappa \notin \Delta$ and $\iota : \Delta \to \Delta, \kappa$ the inclusion*

(i) *If $n \leq m$ and $(E,\delta) \leq (E',\delta')$ then $\iota^!_n(E,\delta) \leq \iota^!_m(E',\delta')$.*

(ii) *For any $(E,\delta) \in \mathfrak{I}(\Delta,\kappa)$ we have $(E,\delta) \leq \iota^!_{\delta(\kappa)}(\mathfrak{I}(\iota)(E,\delta))$.*

(iii) *For any $(E,\delta) \in \mathfrak{I}(\Delta)$ and any $n \in \mathbb{N}$ we have $\mathfrak{I}(\iota)\left(\iota^!_n(E,\delta)\right) = (E,\delta)$.*

(iv) *For any $(E,\delta) \in \mathfrak{I}(\Delta,\kappa)$ and $\kappa' \in \Delta$, ${\delta_n^!}^{-\kappa'} = \left(\delta^{-\kappa'}\right)^!_n$.*

We are now ready to describe the right adjoint $\forall\kappa$ to ι^*. Let Δ be a clock context, κ a clock not in Δ and $\iota : \Delta \to \Delta, \kappa$ the inclusion.

Define $\forall\kappa : \mathfrak{GR}(\Delta,\kappa) \to \mathfrak{GR}(\Delta)$ on an object $X \in \mathfrak{GR}(\Delta,\kappa)$ at stage $(E,\delta) \in \mathfrak{I}(\Delta)$ by taking the limit (in Set) of the diagram of restrictions

$$X\left(\iota^!_1(E,\delta)\right) \leftarrow X\left(\iota^!_2(E,\delta)\right) \leftarrow X\left(\iota^!_3(E,\delta)\right) \leftarrow \cdots$$

where the arrows are X's restrictions using Lemma 3.16. The restrictions of $\forall\kappa.(X)$ and the action of $\forall\kappa$ on morphisms are determined purely formally from the universal properties of limits. The unit η of the adjunction is constructed using the universal property of the limit using Lemma 3.16.(iii) which shows that the diagram

$$\iota^*(X)\left(\iota^!_1(E,\delta)\right) \leftarrow \iota^*(X)\left(\iota^!_2(E,\delta)\right) \leftarrow \iota^*(X)\left(\iota^!_3(E,\delta)\right) \leftarrow \cdots \tag{2}$$

is a constant diagram. The counit ε is constructed with the projections of the limit together with Lemma 3.16.(ii). In more detail, $\varepsilon^X : \iota^*(\forall\kappa(X)) \to X$ and so at stage (E,δ) we must define a function $\varepsilon^X_{(E,\delta)} : \iota^*(\forall\kappa(X))(E,\delta) \to X(E,\delta)$ which is a function from the limit of

$$X\left(\iota^!_1\left(\mathfrak{I}(\iota)(E,\delta)\right)\right) \leftarrow X\left(\iota^!_2\left(\mathfrak{I}(\iota)(E,\delta)\right)\right) \leftarrow \cdots \leftarrow X\left(\iota^!_{\delta(\kappa)}\left(\mathfrak{I}(\iota)(E,\delta)\right)\right) \leftarrow \cdots$$

to $X(E,\delta)$. There is a projection from the limit to $X\left(\iota^!_{\delta(\kappa)}\left(\mathfrak{I}(\iota)(E,\delta)\right)\right)$ and from Lemma 3.16.(ii) we have $(E,\delta) \leq \iota^!_{\delta(\kappa)}\left(\mathfrak{I}(\iota)(E,\delta)\right)$ which means there is a function

$$X\left((E,\delta) \leq \iota^!_{\delta(\kappa)}\left(\mathfrak{I}(\iota)(E,\delta)\right)\right) : X\left(\iota^!_{\delta(\kappa)}\left(\mathfrak{I}(\iota)(E,\delta)\right)\right) \to X(E,\delta).$$

Since the diagram is in Set we could describe the limit very explicitly as the set of compatible sequences. This is useful for checking some properties, but we omit it here due to lack of space.

Equipped with a this description of $\forall\kappa$ we are able to show the necessary properties for interpreting the rules of the type theory.

Proposition 3.17 (Properties of $\forall\kappa$) *Let Δ be a clock context and $\kappa \in \mathrm{CV}$ a clock not in Δ. The functor $\forall\kappa$ satisfies*

(i) *The unit η of the adjunction $\iota^* \dashv \forall\kappa$ is a natural isomorphism. Hence ι^* is a full and faithful functor witnessing that $\mathfrak{GR}(\Delta)$ is a full subcategory of $\mathfrak{GR}(\Delta, \kappa)$.*

(ii) *The functor $\forall\kappa$ preserves all coproducts, but not colimits in general.*

(iii) *For any object $X \in \mathfrak{GR}(\Delta, \kappa)$ the canonical morphism $c : \forall\kappa.X \to \forall\kappa.(\blacktriangleright^\kappa X)$ defined as $c = \forall\kappa.(\mathsf{next}^\kappa)$ is an isomorphism.*

(iv) (Beck-Chevalley condition for $\forall\kappa$) *Let $f : \Delta_1 \to \Delta_2$ be a function between two clock contexts, and let $\kappa \notin \Delta_1 \cup \Delta_2$ be a clock. Let and $\iota_1 : \Delta_1 \to \Delta_1, \kappa$ and $\iota_2 : \Delta_2 \to \Delta_2, \kappa$ be the two inclusions.*

For every $X \in \mathfrak{GR}(\Delta_1, \kappa)$ the presheaves $f^*(\forall\kappa.X)$ and $\forall\kappa.(f + id_\kappa)^*(X)$ are equal *and the canonical morphism $\forall\kappa.((f + id_\kappa)^*(\varepsilon)) \circ \eta^{f^*(\forall\kappa.X)}$ from $f^*(\forall\kappa.X)$ to $\forall\kappa.(f + id_\kappa)^*(X)$ is the identity.*

(v) *Let Δ be a clock context, $\kappa' \in \Delta$, $\kappa \notin \Delta$ and $X \in \mathfrak{GR}(\Delta, \kappa)$ the canonical morphism $\forall\kappa.(\blacktriangleright^{\kappa'}(\varepsilon)) \circ \eta : \blacktriangleright^{\kappa'}(\forall\kappa.X) \to \forall\kappa.\blacktriangleright^{\kappa'} X$ is an isomorphism.*

Proof sketch

(i) Using Lemma 3.16.(iii) the object $\forall\kappa.\iota^*(X)$ at stage (E, δ) is the limit of the constant diagram (2). Because the diagram is connected its limit is isomorphic to $X(E, \delta)$ by the unique mediating map, which is by definition the unit η. The second part is a standard fact about adjoint functors [12, Theorem IV.3.1].

(ii) The reason this property holds is that coproducts are given pointwise and that in Set coproducts commute with connected limits.

(iii) The arrow $\forall\kappa.(\mathsf{next}^\kappa)$ at stage $(E, \delta) \in \mathfrak{I}(\Delta)$ is by definition the mediating map from the limit of

$$X\left(\iota_1^!(E, \delta)\right) \leftarrow X\left(\iota_2^!(E, \delta)\right) \leftarrow X\left(\iota_3^!(E, \delta)\right) \leftarrow \cdots$$

to the limit of

$$1 \leftarrow X\left(\iota_1^!(E, \delta)\right) \leftarrow X\left(\iota_2^!(E, \delta)\right) \leftarrow X\left(\iota_3^!(E, \delta)\right) \leftarrow \cdots$$

so it is an isomorphism.

(iv) The proof is somewhat technical due to the amount of notation involved, but essentially straightforward. Lemma 3.16 is used.

(v) Follows by computation and Lemma 3.16.(iv). Note that to even state it Proposition 3.14 is used to get $\iota^* \circ \blacktriangleright^{\kappa'} = \blacktriangleright^{\kappa'} \circ \iota^*$ so we could apply the counit ε.

□

Extension of $\forall\kappa$ to slices proceeds exactly as before [16, Proposition 1]. The interpretation of the clock instantiation $t[\kappa']$ now proceeds as follows. A term $\Delta \mid \Gamma \vdash t : \forall\kappa.\tau$ corresponds to a morphism from (the interpretation of) Γ to $\forall\kappa.\tau$ in $\mathfrak{GR}(\Delta)$. Transposing along the adjunction $\iota^* \dashv \forall\kappa$ we get a morphism t' from $\iota^*(\Gamma)$ to τ in $\mathfrak{GR}(\Delta, \kappa)$. Let $f : \Delta, \kappa \to \Delta$ be the identity on Δ and map κ to κ'. Applying $\mathfrak{GR}(f)$ to t' we get a morphism from $\mathfrak{GR}(f)(\mathfrak{GR}(\iota)(\Gamma))$ to $\mathfrak{GR}(f)(\tau)$ in $\mathfrak{GR}(\Delta)$ which we define to be the interpretation of $t[\kappa']$. Notice that

$\mathfrak{GR}(f)(\mathfrak{GR}(\iota)(\Gamma))$ is just Γ and by definition $\mathfrak{GR}(f)(\tau)$ is the interpretation of $\tau[\kappa'/\kappa]$, so the interpretation is consistent.

Remark 3.18 This interpretation is standard, see e.g. Jacobs [10], but note that it is crucial that we have general clock substitution $\mathfrak{GR}(f)$, for arbitrary f, and this is precisely the ingredient that was missing in previous models, hence the restrictions on clock instantiation rules.

3.5 Universes

We follow previous work [5,16] and use Hofmann and Streicher's construction of universes in presheaf toposes from universes in Set [9] which we now recall instantiated to our special case. We first recall what a semantic universe is.

Definition 3.19 Let $\mathbb{C}$ be a locally cartesian closed category with coproducts and $\mathsf{el} : E \to U$ a morphism in $\mathbb{C}$. A morphism $f : A \to \Gamma$ is *small* with respect to el if there is a morphism $\overline{f} : \Gamma \to U$ such that f is appears as the pullback of el along $\overline{f}$. The morphism $\overline{f}$ is called a *code* of f. An object Γ is small if the unique map $\Gamma \to 1$ is small.

The map el is a *universe* if the objects 0, 1, $\mathbb{N}$ are small and the notion of smallness is closed under composition, finite coproducts and small dependent products.

Let $\mathbb{U}$ be a Grothendieck universe in Set such that $\mathbb{N} \in \mathbb{U}$ and let Δ be a finite set of clocks.

Definition 3.20 The presheaf $V^\Delta \in \mathfrak{GR}(\Delta)$ is defined as $V^\Delta(E,\delta) = \mathbb{U}^{\downarrow(E,\delta)^{\mathrm{op}}}$ where $\downarrow(E,\delta)^{\mathrm{op}}$ is the set of elements of $\mathfrak{I}(\Delta)$ below (E,δ) and $\mathbb{U}^{\downarrow(E,\delta)^{\mathrm{op}}}$ is the set of presheaves D on $\downarrow(E,\delta)^{\mathrm{op}}$ such that for all $(E',\delta') \le (E,\delta)$ we have $D(E',\delta') \in \mathbb{U}$.

The action of V^Δ on morphisms is by precomposition: $V^\Delta((E_1,\delta_1) \le (E_2,\delta_2))(D) = D \circ \iota$ where ι is the inclusion of $\downarrow(E_1,\delta_1)$ to $\downarrow(E_2,\delta_2)$.

The presheaf of elements E^V_Δ is defined as $E^V_\Delta(E,\delta) = \sum_{D \in V^\Delta(E,\delta)} D(E,\delta)$ with restrictions $E^V_\Delta((E_1,\delta_1) \le (E_2,\delta_2))(D,x) = (D \circ \iota, D((E_1,\delta_1) \le (E_2,\delta_2))(x))$.

The universe is the first projection $u^\Delta : E^V_\Delta \to V^\Delta$ defined as $u^\Delta_{E,\delta}(D,x) = D$.

Hofmann and Streicher [9] show that the universe u^Δ is closed under the usual constructs used to model dependent type theory, provided $\mathbb{U}$ is. What remains is to show that they are also closed under $\forall\kappa$ and $\blacktriangleright^\kappa$ and that they are suitably preserved by reindexing functors $\mathfrak{GR}(f)$. The first two of these properties follow exactly as before [16] so we focus on the last.

The functors $\mathfrak{GR}(f)$ do not in general preserve the universes. In particular the inclusion $\iota^* : \mathfrak{GR}(\Delta) \to \mathfrak{GR}(\Delta,\kappa)$ does not map V^Δ to (an object isomorphic to) $V^{\Delta,\kappa} \in \mathfrak{GR}(\Delta,\kappa)$. However *surjective* substitutions do preserve universes in the appropriate sense.

Lemma 3.21 *Let $s : \Delta \to \Delta'$ be a* surjective *function between clock contexts Δ and Δ'. There exist* natural isomorphisms $c^V : s^*\left(V^\Delta\right) \to V^{\Delta'}$ *and* $c^E : s^*\left(E^V_\Delta\right) \to E^V_{\Delta'}$

such that the diagram

$$\begin{array}{ccc} s^*\left(E^V_\Delta\right) & \xrightarrow{c^E} & E^V_{\Delta'} \\ {\scriptstyle s^*(u^\Delta)}\downarrow & & \downarrow{\scriptstyle u^{\Delta'}} \\ s^*\left(V^\Delta\right) & \xrightarrow{c^V} & V^{\Delta'} \end{array}$$

commutes.

Proof sketch From Lemma 3.4 we have $\mathfrak{I}(s)$ *injective* and from Lemma 3.7 we have that $\mathfrak{I}(s)$ is a fibration. Thus $\mathfrak{I}(s)$ restricted to a function $\downarrow(E,\delta) \to \downarrow\mathfrak{I}(s)(E,\delta)$ is a bijection with an order preserving inverse given by the assignment $u((E,\delta),-)$.

Moreover, because the bijection is given by a restriction of a single function $\mathfrak{I}(s)$ it is natural in (E,δ). We thus have

$$s^*(V^\Delta)(E,\delta) = V^\Delta\left(\mathfrak{I}(s)(E,\delta)\right) = \mathbb{U}^{\downarrow \mathfrak{I}(s)(E,\delta)^{\mathrm{op}}} \cong \mathbb{U}^{\downarrow(E,\delta)^{\mathrm{op}}} = V^{\Delta'}(E,\delta)$$

where the bijection $\mathbb{U}^{\downarrow \mathfrak{I}(s)(E,\delta)^{\mathrm{op}}} \cong \mathbb{U}^{\downarrow(E,\delta)^{\mathrm{op}}}$ is natural in (E,δ). Thus $s^*(V^\Delta) \cong V^{\Delta'}$ as presheaves in $\mathfrak{GR}(\Delta')$. The map c^E is defined similarly. □

Remark 3.22 Inspection of the proof also shows why for the inclusion $\iota : \Delta \to \Delta, \kappa$, the reindexing ι^* does not preserve universes in this way. This is consistent with the situation as it was in Møgelberg's previous model [16] and so following *loc. cit.* we add additional universes in each $\mathfrak{GR}(\Delta)$.

Definition 3.23 Let Δ and Δ' be clock contexts such that $\Delta' \subseteq \Delta$. Let $\iota : \Delta' \to \Delta$ be the inclusion. We define the universe $\left(\mathfrak{u}^\Delta_{\Delta'}, \mathcal{E}^\Delta_{\Delta'}, \mathcal{U}^\Delta_{\Delta'}\right)$ as

$$\mathcal{U}^\Delta_{\Delta'} = \iota^*\left(V^{\Delta'}\right) \qquad \mathcal{E}^\Delta_{\Delta'} = \iota^*\left(E^V_{\Delta'}\right) \qquad \mathfrak{u}^\Delta_{\Delta'} = \iota^*\left(u^{\Delta'}\right).$$

Theorem 3.24 *The triple* $\left(\mathfrak{u}^\Delta_{\Delta'}, \mathcal{E}^\Delta_{\Delta'}, \mathcal{U}^\Delta_{\Delta'}\right)$ *is a universe closed under dependent products, sums,* $\forall\kappa$ *and* $\blacktriangleright^\kappa$.

Proof sketch To see that the notion of smallness is closed under dependent product and sum one uses the fact that ι^* is an LCC functor (Theorem 3.8) and the fact that a universe is closed under dependent products if a particular *generic* map is small and this generic map can be constructed using only the LCC structure, hence it is preserved by ι^*. The same approach works for dependent sums. See Shulman [19] for details on how the generic maps for dependent products and sums are constructed.

Closure under $\blacktriangleright$ follows by first showing that the universes V^Δ have codes $\triangleright^\kappa$ for $\blacktriangleright^\kappa$ and then deriving codes for $\mathcal{U}^\Delta_{\Delta'}$ from these using Proposition 3.14. Closure under $\forall\kappa$ is also shown first for universes V^Δ and then using the Beck-Chevalley condition (Proposition 3.17) for $\mathcal{U}^\Delta_{\Delta'}$. See Møgelberg [16] for more details. □

Finally, these additional universes are preserved by clock substitution in the appropriate way.

Proposition 3.25 *Let* $f : \Delta_1 \to \Delta_2$ *be a function between clock contexts* Δ_1 *and* Δ_2. *Let* $\Delta' \subseteq \Delta_1$ *be another clock context and* $\left(\mathfrak{u}^{\Delta_1}_{\Delta'}, \mathcal{E}^{\Delta_1}_{\Delta'}, \mathcal{U}^{\Delta_1}_{\Delta'}\right)$ *the universe from*

Definition 3.23. There exist two natural isomorphisms c_E^f *and* c_V^f *such that the diagram*

$$\begin{array}{ccc} f^*\left(\mathcal{E}^{\Delta_1}_{\Delta'}\right) & \xrightarrow{c_E^f} & \mathcal{E}^{\Delta_2}_{f[\Delta']} \\ {\scriptstyle f^*\left(\mathrm{u}^{\Delta_1}_{\Delta'}\right)}\downarrow & & \downarrow{\scriptstyle \mathrm{u}^{\Delta_2}_{f[\Delta']}} \\ f^*\left(\mathcal{U}^{\Delta_1}_{\Delta'}\right) & \xrightarrow{c_V^f} & \mathcal{U}^{\Delta_2}_{f[\Delta']} \end{array}$$

commutes. In particular, $f^*\left(\mathcal{U}^{\Delta_1}_{\Delta'}\right) \cong \mathcal{U}^{\Delta_2}_{f[\Delta']}$.

Proof Let ι_1 be the inclusion of Δ' into Δ_1 and ι_2 the inclusion of $f[\Delta']$ into Δ_2. Let $s : \Delta' \to f[\Delta']$ be the restriction of f. By definition s is *surjective* and $f \circ \iota_1 = \iota_2 \circ s$ and so $f^* \circ \iota_1^* = \iota_2^* \circ s^*$. Lemma 3.21 gives natural isomorphisms c^V and c^E such that the diagram on the left

$$\begin{array}{ccc} s^*\left(E^V_{\Delta'}\right) & \xrightarrow{c^E} & E^V_{f[\Delta']} \\ {\scriptstyle s^*\left(u^{\Delta'}\right)}\downarrow & & \downarrow{\scriptstyle u^{f[\Delta']}} \\ s^*\left(V^{\Delta'}\right) & \xrightarrow{c^V} & V^{f[\Delta']} \end{array} \qquad \begin{array}{ccc} \iota_2^*\left(s^*\left(E^V_{\Delta'}\right)\right) & \xrightarrow{\iota_2^*\left(c^E\right)} & \iota_2^*\left(E^V_{f[\Delta']}\right) \\ {\scriptstyle \iota_2^*\left(s^*\left(u^{\Delta'}\right)\right)}\downarrow & & \downarrow{\scriptstyle \iota_2^*\left(u^{f[\Delta']}\right)} \\ \iota_2^*\left(s^*\left(V^{\Delta'}\right)\right) & \xrightarrow{\iota_2^*(c^V)} & \iota_2^*\left(V^{f[\Delta']}\right) \end{array}$$

commutes. Hence the diagram on the right commutes and the vertical morphisms are isomorphisms. But notice that, e.g. $\iota_2^*\left(s^*\left(V^{\Delta'}\right)\right) = f^*\left(\iota_1^*\left(V^{\Delta'}\right)\right) = f^*\left(\mathcal{U}^{\Delta_1}_{\Delta'}\right)$ and also by definition $\iota_2^*\left(V^{f[\Delta']}\right) = \mathcal{U}^{\Delta_2}_{f[\Delta']}$. This concludes the proof. □

4 Conclusions and future work

We have sketched (up to solving the coherence problem) that allowing clock synchronisation retains soundness by constructing a model which validates it. With regards to the coherence problem, we can certainly solve it in each $\mathfrak{GR}(\Delta)$, so that substitution of *terms* into types and terms behaves correctly. However we also need to interpret *clock* substitution, which we do using the functors $\mathfrak{GR}(f)$ for functions $f : \Delta_1 \to \Delta_2$ between clock contexts. And in order to validate equalities such as $[\![\Delta_2 \vdash f(\Gamma)]\!] = \mathfrak{GR}(f)([\![\Delta_1 \vdash \Gamma]\!])$ we would require $\mathfrak{GR}(f)$ to preserve our choice of interpretation of all the constructs *on the nose*, but it only does so up to canonical isomorphism.

We believe this is a technical, rather than essential, problem with the particular presentation. In particular, without universes, we do have a solution to the coherence problem by replacing the categories $\mathfrak{GR}(\Delta)$ by equivalent ones obtained by the Bénabou construction [4] (see also [10, Corollary 5.2.5]). This then allows us to make choices of structure that are preserved on the nose by functors interpreting clock substitution. However doing this breaks type equalities like $\mathsf{El}(\mathsf{in}\, t) \simeq \mathsf{El}(t)$ where in is a universe inclusion from $\mathcal{U}^{\Delta}_{\Delta'}$ to $\mathcal{U}^{\Delta}_{\Delta}$, for instance. The types on the left and right are only interpreted as isomorphic objects, not equal.

We are working on giving computational meaning to various type isomorphisms validated by the model and required for working with coinductive types via guarded

recursive types. Removing the "freshness" requirements in clock instantiation rule considerably simplifies the syntactic theory.

Acknowledgement

We thank Lars Birkedal for helpful discussions. Aleš Bizjak is supported in part by a Microsoft Research PhD grant.

References

[1] Abel, A. and B. Pientka, *Wellfounded recursion with copatterns: A unified approach to termination and productivity*, in: *Proceedings ICFP 2013* (2013), pp. 185–196.

[2] Abel, A., B. Pientka, D. Thibodeau and A. Setzer, *Copatterns: Programming infinite structures by observations*, in: *Proceedings of POPL*, POPL '13 (2013), pp. 27–38.

[3] Atkey, R. and C. McBride, *Productive coprogramming with guarded recursion*, in: *Proceedings of ICFP 2013* (2013), pp. 197–208.

[4] Bénabou, J., *Fibrations petites et localement petites*, C. R. Acad. Sci. Paris Sér. A-B **281** (1975), pp. Ai, A897–A900.

[5] Birkedal, L. and R. E. Møgelberg, *Intensional type theory with guarded recursive types qua fixed points on universes*, in: *LICS*, 2013, pp. 213–222.

[6] Birkedal, L., R. E. Møgelberg, J. Schwinghammer and K. Støvring, *First steps in synthetic guarded domain theory: step-indexing in the topos of trees*, Logical Methods in Computer Science **8** (2012).

[7] Clouston, R., A. Bizjak, H. B. Grathwohl and L. Birkedal, *Programming and reasoning with guarded recursion for coinductive types*, in: *Proceedings of FoSSaCS*, 2015, pp. 407–421.

[8] Girard, J.-Y., P. Taylor and Y. Lafont, "Proofs and Types," Cambridge University Press, New York, NY, USA, 1989.

[9] Hofmann, M. and T. Streicher, *Lifting Grothendieck universes* (1999), unpublished.
URL www.mathematik.tu-darmstadt.de/~streicher/NOTES/lift.pdf

[10] Jacobs, B., "Categorical Logic and Type Theory," Number 141 in Studies in Logic and the Foundations of Mathematics, North Holland, Amsterdam, 1999.

[11] Johnstone, P. T., "Sketches of an elephant: a topos theory compendium. Vol. 2," Oxford Logic Guides **44**, The Clarendon Press Oxford University Press, Oxford, 2002, i–xxii, 469–1089 and I1–I71 pp.

[12] MacLane, S., "Categories for the Working Mathematician," Graduate Texts in Mathematics, Springer New York, 1998, second edition.

[13] MacLane, S. and I. Moerdijk, "Sheaves in Geometry and Logic: A First Introduction to Topos Theory," Mathematical Sciences Research Institute Publications, Springer New York, 1992.

[14] The Coq development team, "The Coq proof assistant reference manual," LogiCal Project (2004), version 8.0.
URL http://coq.inria.fr

[15] McBride, C. and R. Paterson, *Applicative programming with effects*, J. Funct. Programming **18** (2008), pp. 1–13.

[16] Møgelberg, R. E., *A type theory for productive coprogramming via guarded recursion*, in: *Proceedings of CSL-LICS 2014* (2014), pp. 71:1–71:10.

[17] Nakano, H., *A modality for recursion*, in: *Proceedings of LICS 2000* (2000), pp. 255–266.

[18] Norell, U., "Towards a practical programming language based on dependent type theory," Ph.D. thesis, Chalmers University of Technology (2007).

[19] Shulman, M., *Univalence for inverse diagrams and homotopy canonicity*, Mathematical Structures in Computer Science **25** (2015), pp. 1203–1277.
URL http://journals.cambridge.org/article_S0960129514000565